Convolutional Codes for *M*-ary Orthogonal and Simplex Channels

R. F. Lyon
Communications Systems Research Section

Convolutional codes chosen for greatest free distance or lowest error probability on a binary channel are not necessarily good codes for an M-ary channel. A rate k/ν coder generates a 2^ν-ary output symbol for each k input bits. If a binary channel is used, the appropriate measure of distance between these symbols is Hamming distance (number of bit disagreements). But if either a 2^ν-ary orthogonal channel, or a 2^ν-ary simplex channel is used, the distance between any two different symbols is unity (number of symbol disagreements). Other distance measures are appropriate on other M-ary channels. Good rate 1/2 and 1/3 codes have been found by computer search for the orthogonal 4-ary and 8-ary channels. The result is a reduction of error probabilities by about a factor of two below previously tabulated codes. The computer technique used is described. At a fixed constraint length, further performance improvement results from increasing ν, up to a limit at $\nu = K$ (constraint length) on the orthogonal 2^ν-ary channel.

I. Introduction

Good codes have codewords that are "far apart" in signal space. When a codeword is a sequence of antipodal signals designed for a binary channel, the Hamming distance defines "far apart." Convolutional codes with the largest possible minimum Hamming distance between any pair of codewords (free distance) have been tabulated for rate 1/2, 1/3, and 1/4, for constraint lengths up to 14 (Ref. 1). But Hamming distance is not the correct measure of distance in many signal spaces; those systems that use a larger number of symbols, $M > 2$, have a wide variety of distance measures. The simplest of these are the orthogonal and simplex distance measures—the distance from a symbol to any other symbol is one, and the distance to itself is zero. Other possibilities are biorthogonal, where the distance from a symbol to its negative is twice all other distances; phase shift keying, where two dimensional Euclidian distances between points on a circle are used; and combined phase and amplitude keyed systems, with more complicated distance measure. We will concern ourselves only with symmetric systems; that is, the set of distances of all symbols from symbol x_i is independent of i.

II. The Convolutional Coder

A convolutional coder consists of a shift register to hold a short history of the input, and a group of mod-2 adders (parity generators) to compute output symbols as a function of past and present inputs (see Fig. 1). Inputs are shifted into the register in groups of k bits; the register holds K such groups, where K is called the constraint length of the coder. The number of binary symbols generated per input of k bits is termed ν. We extend the concept of the coder to let these ν bits select one of $M = 2^\nu$ channel symbols. We make no attempt to generalize the relation between ν and M.

The coder rate is k/ν bits per binary symbol, or k bits per M-ary symbol; that is, each k input bits result in the output of one $2^\nu = M$-ary channel symbol. Then an input of length L (groups of k bits) causes an output of length $L + K - 1$ (symbols).

III. Distance Between Code Words

Since the coder uses only linear mod-2 adders, an all-zero input corresponds to an all-zero output of binary symbols, which selects a symbol sequence all-x_0. Each codeword corresponding to a length L input (with zeroes before and after) differs from a sequence of x_0's in at most $L + K - 1$ places. Then, with $L = 1$, an immediate upper bound on the minimum distance between codewords is

$$d_f \leq K \cdot d_{\max} \tag{1}$$

where $d_{\max}$ is the greatest distance from x_0 to another codeword. This bound is often quite loose. For example, using $M = 4$ with distances 0, 1, 1, 2 corresponds to the familiar Hamming distance on a binary channel with symbols grouped in pairs. Since $d_{\max} = 2$, our simple bound is

$$d_f \leq 2K$$

A much better well-known bound (Ref. 2) for a binary channel is

$$d_f \leq \min_{L>0} \left[\frac{2^L}{2^L - 1} (K + L - 1) \right] \tag{2}$$

For example, at $K = 8$ the bounds are 16 and 10, respectively.

We can informally estimate d_f by assuming the coder generates K outputs which are random and equally likely to be either (a) any of M symbols, or (b) any of $M - 1$ symbols different from x_0. That is,

$$\left.\begin{aligned} &\text{(a)} \quad d_f \approx K \cdot \left[\sum_{i=0}^{M} |x_0 - x_i| \right] \Big/ M \\ &\text{(b)} \quad d_f \approx K \cdot \left[\sum_{i=1}^{M} |x_0 - x_i| \right] \Big/ (M - 1) \end{aligned}\right\} \tag{3}$$

Then in our example ($K = 8$, distances 0, 1, 1, 2) we would estimate

$$\text{(a)} \quad d_f \approx \frac{0 + 1 + 1 + 2}{4} \cdot K = 8$$

$$\text{(b)} \quad d_f \approx \frac{1 + 1 + 2}{3} \cdot K = 10.7$$

We conjecture that for any symmetric distance measure, and large enough K, (3b) is an upper bound on d_f, though for small K (e.g., $K = 1$) it is obvious that the greater bound $d_f \leq K \cdot d_{\max}$ can be achieved. We further conjecture that a d_f at least as great as (3a) can always be achieved, even for large K.

IV. Finding Codes With Large Free Distance

In the last section we showed how to estimate the free distance of a good code. In this section we describe a computer technique for finding codes that meet or nearly meet this estimate. We will consider only orthogonal and simplex symbols—all distances are one, and the estimate (3b) is the same as the bound (1); $d_f \leq K$.

We will represent the connections from the shift register to the adders by ν vectors of kK bits; these are called connection vectors $g_1, g_2, \cdots, g_\nu$. A one in bit i of g_j represents a connection from the ith shift register stage, $1 \leq i \leq kK$, to the jth adder. The output of the jth adder after the nth bit of data input $x(n)$ is

$$S_j(n) = g_j * x = \sum_{i=1}^{kK} g_j(i) \cdot x(n - i + 1) \pmod 2 \tag{4}$$

The data are convolved with the connection vectors (code generators or generating polynomials), hence the name convolutional code.

To find good codes of short constraint length, we propose to test all possible values of the connection vectors.

For $K = 8$, $k = 1$, $\nu = 2$ (code rate $R_c = 1/2$), the number of possibilities is

$$N = 2^{K\nu} = 2^{16} = 65536$$

However, many of these are duplications. We may wish to check each possibility to see if its reflections have already been considered before proceeding to evaluate the free distance and other properties.

An obvious way to measure free distance is to compute the distance of all codewords from the zero codeword, and take the minimum. This procedure is not effective, however, because there are an infinite number of codewords to test. We can modify the procedure to check distances of all codewords generated by inputs of length $\ell \leq L_{\max}$. But then we have to consider what value of $L_{\max}$ is sufficient; a proven bound on the sufficient $L_{\max}$ is prohibitively large (Ref. 3). But experience shows that $L_{\max} = kK$ (bits) is probably sufficient (at least for orthogonal distance measure). Then there are $2^{L_{\max}-1} = 128$ inputs to consider for each code in our example. For all codes, the total number of inputs to check is $2^{K\nu + L_{\max}-1} = 2^{23}$. Each requires the calculation of up to $\nu(kK + L_{\max} - 1) = 30$ output bits, by adding $kK = 8$ bits modulo 2. This makes about $2^{31} \approx 2$ billion elementary operations. For $K = 12$ this figure becomes about $2^{44} \approx 10^{13}$.

But all is not hopeless. Suppose we wish to search for codes with $d_f \geq d_{\min}$. Use a program that starts checking with short inputs, and gives up on a code as soon as any distance is less than $d_{\min}$. For example, we hope to meet the bound $d_f = 8$ in the $K = 8$ example, so set $d_{\min} = 8$. The first input to try is a single one ($\cdots 010 \cdots$). The output will just be K groups of ν bits, one from each connection vector. If any group of ν bits is all zero, the distance will not equal K, otherwise it will. So there are $(2^\nu - 1)^K$ codes that pass this first step, out of $2^{\nu K}$ total tested. In our example, 6561 codes survive (about 10%). In $2^{\nu K} \cdot kK \cdot kK = 2^{22} = 4$ million elementary operations, the bulk of the task has been eliminated. Less dramatic reductions are made for successively longer inputs. To eliminate a large memory requirement, the loop on input lengths should be within the loop on codes. Any code that survives through $\ell = L_{\max}$ is printed. Comparisons can then be made between the codes generated to determine which are better by some other criteria.

Rather than continuing the procedure through a large $L_{\max}$ which is felt to be sufficient, it may be useful to stop with a moderately small $L_{\max}$, and send the surviving codes to a true free distance computing algorithm; several effective algorithms are known (Refs. 1, 4, and 5).

V. Evaluating Other Code Properties

Associated with every convolutional code is a transfer function[1] (Ref. 6)

$$T(D, N, L) = \sum_{i=1}^{\infty} \sum_{j=1}^{\infty} \sum_{\ell=1}^{j} a_{ij\ell} D^i N^j L^\ell \tag{5}$$

where each term represents $a_{ij\ell}$ inputs of length ℓ, with codewords at distance i from the all-zero codeword, and differing from the all-zero input in j bits. The transfer function is useful for evaluating bounds on the first event error probability P_E and the bit error probability P_B (Ref. 6). It is shown that

$$\begin{aligned} P_E &\leq T(D, N, L)\big|_{N=1,\, L=1,\, D=D_0} \\ P_B &\leq \left.\frac{\partial T(D, N, L)}{\partial N}\right|_{N=1,\, L=1,\, D=D_0} \end{aligned} \tag{6}$$

where for a stationary channel

$$D_0 = \sum_{y_r} P(y_r|x_r)^{1/2} P(y_r|x_r')^{1/2} < 1 \tag{7}$$

for $\{y_r\}$ the set of channel outputs, and x_r and x_r' the correct and incorrect input symbols, separated by a unit distance.

The computer procedure described in the preceding section is useful in that all coefficients of terms of $T(D, N, L)$ with exponents of L less than or equal to $L_{\max}$ can be exactly calculated with little extra computation. Then P_E and P_B can be estimated, though not necessarily bounded.

Since there are several codes with a maximum value of d_f, we must choose between them by other considerations, such as: (a) minimum P_E as $D_0 \to 0$; (b) minimum P_B as $D_0 \to 0$; (c) minimum P_E at some chosen value of D_0; (d) minimum P_B at some chosen value of D_0. It is often possible to find a code which is best or almost best on all these conditions over a wide range of D_0.

FORTRAN subroutines, listed in the appendix, have been written to carry out the evaluation of $T(D, N, L)$, and P_E and P_B, for a list of values of D_0. The subroutine

[1]Differs from Viterbi's representation in the meaning of the exponent of L.

EVAL is set up to return without output as soon as any distance is less than the supplied value of M, or to return the new d_f in variable M if $d_f \geq M$; so in checking a list of possible codes, M can be initially set small (even zero) and the program will print all codes with d_f at least as great as the d_f of the previously printed code.

VI. Results

For $K \leq 8$ there are many rate 1/2 codes with $d_f = K$ on the 4-ary orthogonal channel. For $K \geq 9$, there are none with $d_f = K$.

Table 1 lists the best codes found (in terms of P_B for $D_0 < 0.4$), along with some previously known codes (designed for the binary channel). The leading terms of the error probability estimates are also listed in Table 1. Figure 2 is a graph of the estimates of P_B versus D_0 for all the codes in Table 1. Notice that the previously known $K = 11$ code is worse than either of the $K = 10$ codes.

Codes with $d_f = K$ are much more abundant for $\nu > 2$; but the number of codes to check is also much greater. At $K = 8$, $\nu = 3$ (rate 1/3) there are $2^{\nu K} = 16$ million codes to check; the first distance check with input $\cdots 010 \cdots$ only brings this down to $(2^\nu - 1)^K = 5.8$ million. But we can find plenty of good codes without checking these by building on the rate 1/2 codes. If g_1 and g_2 form a rate 1/2 code with free distance d_f, then the addition of any g_3 forms a rate 1/3 code with free distance $\geq d_f$ on the 8-ary channel. If the bound was met at $\nu = 2$, it will be met at $\nu = 3$; if it was not met at $\nu = 2$, it may be improved at $\nu = 3$. And many good rate 1/3 codes will also be good rate 2/3 codes. Similarly, good codes are easily found for the 16-ary orthogonal channel at rates 1/4, 2/4, and 3/4.

The rate 1/3 codes in Table 2 were found by taking g_1 and g_2 from the best rate 1/2 code, then trying all values of g_3 to find the best. The resulting bit error probabilities are compared with those of previously tabulated codes (Ref. 1) in Fig. 3. Notice that the best $K = 4$ code found by this procedure is not quite as good as the one from Ref. 1.

We can hypothesize a best possible rate $1/\nu$ code by assuming that every non-zero data sequence in the coder will result in an output different from the zero output. Then the distance of any codeword caused by a length ℓ input ($\ell \leq K$) will be $K + \ell - 1$. Thus the transfer function of the code will be

$$\begin{aligned} T(D,N,L) &= D^K NL + D^{K+1}N^2L^2 + D^{K+2}(N^2 + N^3)L^3 \\ &\quad + D^{K+3}(N^2 + 2N^3 + N^4)L^4 + \cdots \\ &= D^K\left[NL + \sum_{l=2}^{K} D^{l-1}L^l \sum_{j=2}^{l} \binom{l-2}{j-2} N^j \right] \\ &\quad + \text{terms with higher } l \end{aligned} \tag{8}$$

We can continue to estimate P_E and P_B by truncating the transfer function to $l \leq L_{\max} = K$; thus we will have a comparison between our codes and these ideal codes. We see that

$$\begin{aligned} P_E &\approx D_0^K\left[1 + \sum_{l=2}^{K} D_0^{l-1} \sum_{j=2}^{l} \binom{l-2}{j-2} \right] \\ &= D_0^K\left[1 + D_0 \sum_{l=2}^{K} (2D_0)^{l-2} \right] \\ &\approx D_0^K[1 + D_0 + 2D_0^2 + 4D_0^3 + 8D_0^4 + \cdots] \end{aligned} \tag{9}$$

and

$$\begin{aligned} P_B &\approx D_0^K\left[1 + \sum_{l=2}^{K} D_0^{l-1} \sum_{j=2}^{l} j\binom{l-2}{j-2} \right] \\ &= D_0^K\left[1 + \sum_{l=2}^{K} (2D_0)^{l-2} \cdot \left(\frac{l+2}{2}\right) \right] \\ &\approx D_0^K[1 + 2D_0 + 5D_0^2 + 12D_0^3 + 28D_0^4 + \cdots] \end{aligned} \tag{10}$$

For a fixed constraint length K, these codes can actually be achieved if ν is allowed to increase sufficiently (this corresponds to an exponential increase in bandwidth when using an orthogonal signal set of size 2^ν). In fact, these codes are the "orthogonal tree codes" of Ref. 7 with $\nu = K$. Each connection vector has a single one; that is, the ν-bit symbol number is just taken from the last K input bits, with no mod-2 adders needed (see Fig. 4).

Figure 5 shows how close our rate 1/2 and 1/3 codes come to this idealization, on the basis of P_B. Notice that at $K = 3$, $\nu = 3$ these achieve the idealization, but for large K, miss by more than 10% of D_0.

VII. Application to Noncoherent MFSK

A channel with random phase disturbances used with multi-frequency-shift-keying is described in the literature (Refs. 8, 9, and 10). From the channel output statistics

we can calculate D_0 as a function of the predetection signal-to-noise ratio, $ST/N_0 = \alpha^2/2$. When frequency m is transmitted, the receiver outputs are $r_1, r_2, \cdots, r_M$, where

$$p(r_j) = \begin{cases} r_j \exp\left(-\frac{r_j^2}{2}\right), & \text{if} \quad j \neq m \\ r_m \exp\left(-\frac{r_m^2}{2} - \frac{\alpha^2}{2}\right) I_0(\alpha r_m), & \text{if} \quad j = m \end{cases} \tag{11}$$

Then we can calculate D_0 from Eq. (7):

$$\begin{aligned} D_0 &= \int\int p(r_j)^{1/2} p(r_m)^{1/2} dr_j dr_m, \qquad j \neq m \\ &= \left\{ \int_0^\infty r \exp\left(-\frac{r^2}{2}\right) \left[\exp\left(-\frac{\alpha^2}{2}\right) I_0(\alpha r) \right]^{1/2} dr \right\}^2 \end{aligned} \tag{12}$$

Notice that D_0 does not depend on M, the number of frequencies (symbols). By using Taylor series we can compute an asymptotic value for small α:

$$D_0 \approx \exp\left(-\frac{\alpha^4}{16}\right) \approx \exp\left[-\left(\frac{ST}{2N_0}\right)^2\right], \qquad \frac{ST}{N_0} \ll 1$$

Using $I_0(x) \to \exp(x)/\sqrt{2\pi x}$, we find for large α:

$$D_0 \approx \frac{\sqrt{\pi}}{2} \alpha \exp\left(-\frac{\alpha^2}{4}\right) = \sqrt{\frac{\pi ST}{2N_0}} \exp\left(-\frac{ST}{2N_0}\right), \qquad \frac{ST}{N_0} \gg 1$$

A numerically calculated curve of D_0 versus ST/N_0 is shown in Fig. 6. When using this channel with small ST/N_0, we will repeat each symbol n times (extend the tone duration to nT), to bring D_0 down to

$$D_0' = D_0^n \approx \exp\left[-n\left(\frac{ST}{2N_0}\right)^2\right], \qquad \frac{ST}{N_0} < 1 \tag{13}$$

We will use a rate $1/\nu$ coder to send one bit per branch, at a bit rate of $1/(nT)$ bps. From a specification of acceptable bit error rate and a chosen code, we can choose D_0, and hence determine the required value of the product $n(ST/2N_0)^2$.

For example, if we need $P_B \leq 10^{-3}$, and we wish to use $M = 4$ frequencies and constraint length $K = 7$, Fig. 2 indicates that $D_0 \leq 0.25$ will suffice. This requires $n(ST/2N_0)^2 \geq 1.39$. Now if we are constrained by power to $ST/N_0 = 0.2$, then we must use $n = 139$ or more. The resulting data rate is $R = 1/139T$ bps, compared to the wideband capacity (Ref. 8)

$$C = \frac{S}{N_0 \ln 2} \cdot \frac{ST/N_0}{2 + (ST/N_0)} = 0.0262/T \text{ bps}$$

for an estimated code efficiency of

$$\frac{R}{C} = \frac{1}{139 \cdot 0.0262} = 0.275$$

(See Ref. 9 for some actual simulated code efficiencies.)

Can we improve on this? Suppose we keep $K = 7$ fixed and allow ν to increase to K, and M to increase to $2^\nu = 128$ frequencies. The ideal code ($\nu = K$) performance from Fig. 5 indicates that $D_0 \leq 0.324$, or $n(ST/2N_0)^2 \geq 1.13$ will suffice. Keeping $ST/N_0 = 0.2$, we need $n = 113$, for $R = 1/113T$, and $R/C = 0.338$. Thus, by allowing bandwidth expansion, without necessitating a more complicated decoder, we improve the rate by $139/113 = 1.22$. Furthermore, the coder is simplified to just a K-bit shift register, with parallel outputs going to an M-ary transmitter. However, the increased bandwidth necessitates a more complicated receiver and demodulator.

VIII. Conclusions

The bounds on P_E and P_B from Ref. 6 do not converge for $D_0 \geq 0.5$, and are very difficult to evaluate in any case. But the estimates presented in this paper, based on error sequences of length $\ell \leq L_{\max}$ are easily formed finite sums. These estimates are useful in finding good convolutional codes, and in predicting error statistics.

Good codes are needed for each different measure of distance on a channel. These good codes can look surprisingly different from codes designed for binary channels—the best code of fixed constraint length for the very noisy wideband noncoherent MFSK channel simply sends each input bit to the transmitter K times in K positions.

More work is needed to compile a list of good codes for all commonly used M-ary channels.

References

1. Larsen, K. J., "Short Convolutional Codes with Maximal Free Distance for Rates 1/2, 1/3, and 1/4," *IEEE Trans. Inform. Theory*, Vol. IT-19, pp. 371–372, May 1973.

2. Heller, J. A., "Sequential Decoding: Short Constraint Length Convolutional Codes," in *Supporting Research and Advanced Development*, Space Programs Summary 37-54, Vol. III, pp. 171–177, Jet Propulsion Laboratory, Pasadena, Calif., Dec. 31, 1968.

3. Costello, D. J., *Construction of Convolutional Codes for Sequential Decoding*, Technical Report EE-692, Department of Elec. Eng., Univ. of Notre Dame, Notre Dame, Ind., Aug. 1969.

4. Forney, G. D., Jr., "Use of a Sequential Decoder to Analyze Convolutional Code Structure," *IEEE Trans. Inform. Theory*, Vol. IT-16, pp. 793–795, Nov. 1970.

5. Bahl, L. R., Cullum, C. D., Frazer, W. D., and Jelinek, F., "An Efficient Algorithm for Computing Free Distance," *IEEE Trans. Inform. Theory*, Vol. IT-18, pp. 437–439, May 1972.

6. Viterbi, A. J., "Convolutional Codes and Their Performance in Communication Systems," *IEEE Trans. Commun. Technol.*, Vol. COM-19, pp. 751–772, Oct. 1971.

7. Viterbi, A. J., "Orthogonal Tree Codes," in *Supporting Research and Advanced Development*, Space Programs Summary 37-39, Vol. IV, pp. 204–209, Jet Propulsion Laboratory, Pasadena, Calif., June 30, 1966.

8. Bar-David, I., and Butman, S., "Performance of Coded, Noncoherent, Hard-Decision MFSK Systems," Technical Report 32-1526, Vol. XIII, pp. 82–91, Jet Propulsion Laboratory, Pasadena, Calif., Nov. 1972.

9. Butman, S., and Klass, M. J., "Capacity of Noncoherent Channels," Technical Report 32-1526, Vol. XVIII, pp. 85–93, Jet Propulsion Laboratory, Pasadena, Calif., Sept. 1973.

10. Butman, S. A., and Lyon, R. F., "Performance of Noncoherent MFSK Channels with Coding," *International Telemetering Conference Proceedings*, Los Angeles, October 15-17, 1974, Vol. X, pp. 142–150.

Table 1. Best known rate 1/2 convolutional codes (a) and previously tabulated codes (b) with error probability estimates based on error sequences with $\ell \leq K$, on the orthogonal 4-ary channel

K	Code (hexadecimal)		P_E estimate	P_B estimate	Type
2	2	3	$D^2 + D^3$	$D^2 + 2D^3$	a, b
3	5	7	$D^3 + 2D^4 + \cdots$	$D^3 + 4D^4 + \cdots$	a, b
4	A	D	$D^4 + 4D^5 + \cdots$	$D^4 + 10D^5 + \cdots$	a
4	D	F	$2D^4 + 3D^5 + \cdots$	$3D^4 + 8D^5 + \cdots$	b
5	12	1F	$2D^5 + 3D^6 + \cdots$	$3D^5 + 7D^6 + \cdots$	a
5	13	1D	$3D^5 + 2D^6 + \cdots$	$6D^5 + 7D^6 + \cdots$	b
6	2E	3D	$2D^6 + 8D^7 + \cdots$	$3D^6 + 25D^7 + \cdots$	a
6	2B	3D	$4D^6 + 8D^7 + \cdots$	$10D^6 + 21D^7 + \cdots$	b
7	52	6D	$4D^7 + 8D^8 + \cdots$	$7D^7 + 27D^8 + \cdots$	a
7	5B	79	$D^6 + 4D^7 + 7D^8 + \cdots$	$D^6 + 10D^7 + 24D^8 + \cdots$	b
8	AD	DF	$6D^8 + 12D^9 + \cdots$	$17D^8 + 49D^9 + \cdots$	a
8	A7	F9	$2D^7 + 3D^8 + 12D^9 + \cdots$	$5D^7 + 6D^8 + 43D^9 + \cdots$	b
9	172	19F	$D^8 + 7D^9 + \cdots$	$2D^8 + 22D^9 + \cdots$	a
9	171	1EB	$2D^8 + 5D^9 + \cdots$	$4D^8 + 16D^9 + \cdots$	b
10	2DD	312	$2D^9 + 8D^{10} + \cdots$	$3D^9 + 37D^{10} + \cdots$	a
10	277	365	$D^8 + 3D^9 + 8D^{10} + \cdots$	$D^8 + 7D^9 + 27D^{10} + \cdots$	b
11	5AD	73F	$3D^{10} + 18D^{11} + \cdots$	$5D^{10} + 70D^{11} + \cdots$	a
11	4DD	7B1	$D^8 + D^9 + 3D^{10} + 20D^{11} + \cdots$	$3D^8 + 2D^9 + 6D^{10} + 95D^{11} + \cdots$	b
12	A4F	DAD	$9D^{11} + 20D^{12} + \cdots$	$27D^{11} + 113D^{12} + \cdots$	a
12	8DD	BD3	$3D^{10} + 8D^{11} + 26D^{12} + \cdots$	$6D^{10} + 42D^{11} + 128D^{12} + \cdots$	b

Table 2. Rate 1/3 convolutional codes and error probability estimates: (a) best codes found by procedure in text, (b) previously known codes

K	Code (hexadecimal)			P_E estimate	P_B estimate	Type
3	5	7	4	$D^3 + D^4 + 2D^5$	$D^3 + 2D^4 + 5D^5$	a
3	5	7	7	$D^3 + 2D^4 + D^5$	$D^3 + 4D^4 + 3D^5$	b
4	A	D	9	$D^4 + D^5 + 3D^6 + \cdots$	$D^4 + 2D^5 + 8D^6 + \cdots$	a
4	B	D	F	$D^4 + D^5 + 3D^6 + \cdots$	$D^4 + 2D^5 + 7D^6 + \cdots$	b
5	12	1F	14	$D^5 + D^6 + 4D^7 + \cdots$	$D^5 + 2D^6 + 11D^7 + \cdots$	a
5	15	1B	1F	$D^5 + 2D^6 + 2D^7 + \cdots$	$D^5 + 4D^6 + 6D^7 + \cdots$	b
6	2E	3D	24	$D^6 + D^7 + 5D^8 + \cdots$	$D^6 + 2D^7 + 13D^8 + \cdots$	a
6	27	2B	3D	$D^6 + 2D^7 + 3D^8 + \cdots$	$D^6 + 5D^7 + 10D^8 + \cdots$	b
7	52	6D	46	$D^7 + 2D^8 + 4D^9 + \cdots$	$D^7 + 4D^8 + 15D^9 + \cdots$	a
7	5B	65	7D	$2D^7 + 2D^8 + 5D^9 + \cdots$	$3D^7 + 7D^8 + 18D^9 + \cdots$	b
8	AD	DF	99	$D^8 + 2D^9 + 7D^{10} + \cdots$	$D^8 + 5D^9 + 22D^{10} + \cdots$	a
8	95	D9	F7	$D^8 + 3D^9 + 5D^{10} + \cdots$	$D^8 + 7D^9 + 15D^{10} + \cdots$	b
9	172	19F	134	$D^9 + 2D^{10} + 8D^{11} + \cdots$	$D^9 + 5D^{10} + 22D^{11} + \cdots$	a
9	16F	1B3	1C9	$D^9 + 2D^{10} + 8D^{11} + \cdots$	$D^9 + 5D^{10} + 26D^{11} + \cdots$	b
10	2DD	312	27B	$D^{10} + 5D^{11} + 4D^{12} + \cdots$	$D^{10} + 14D^{11} + 14D^{12} + \cdots$	a
10	24F	2F5	39B	$D^{10} + 6D^{11} + 3D^{12} + \cdots$	$D^{10} + 19D^{11} + 10D^{12} + \cdots$	b
11	5AD	73F	474	$D^{11} + 5D^{12} + 8D^{13} + \cdots$	$D^{11} + 14D^{12} + 28D^{13} + \cdots$	a
11	56B	5B9	67D	$2D^{11} + 8D^{12} + 4D^{13} + \cdots$	$4D^{11} + 28D^{12} + 15D^{13} + \cdots$	b
12	A4F	DAD	959	$D^{12} + 6D^{13} + 9D^{14} + \cdots$	$D^{12} + 16D^{13} + 37D^{14} + \cdots$	a
12	9F7	BD3	CB5	$D^{11} + 6D^{13} + 9D^{14} + \cdots$	$D^{11} + 16D^{13} + 32D^{14} + \cdots$	b

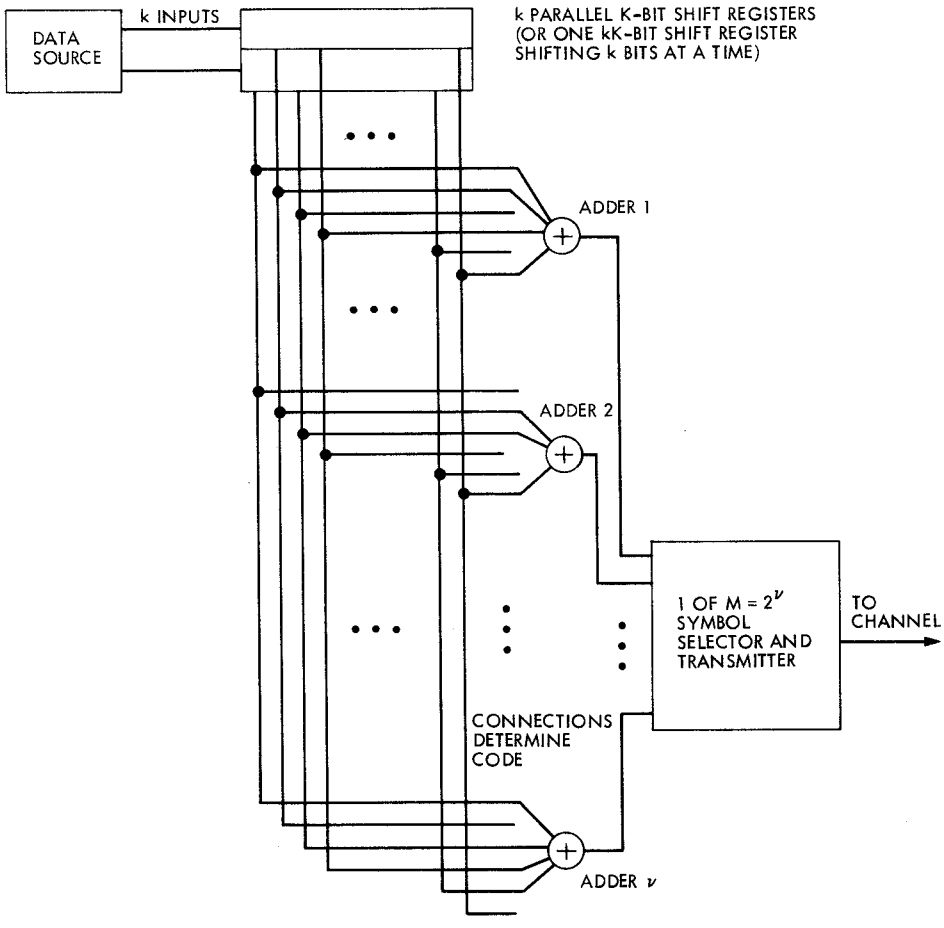

Fig. 1. A rate k/ν convoiutional coder for the M-ary channel with $M = 2^{\nu}$, drawn for $k = 2$

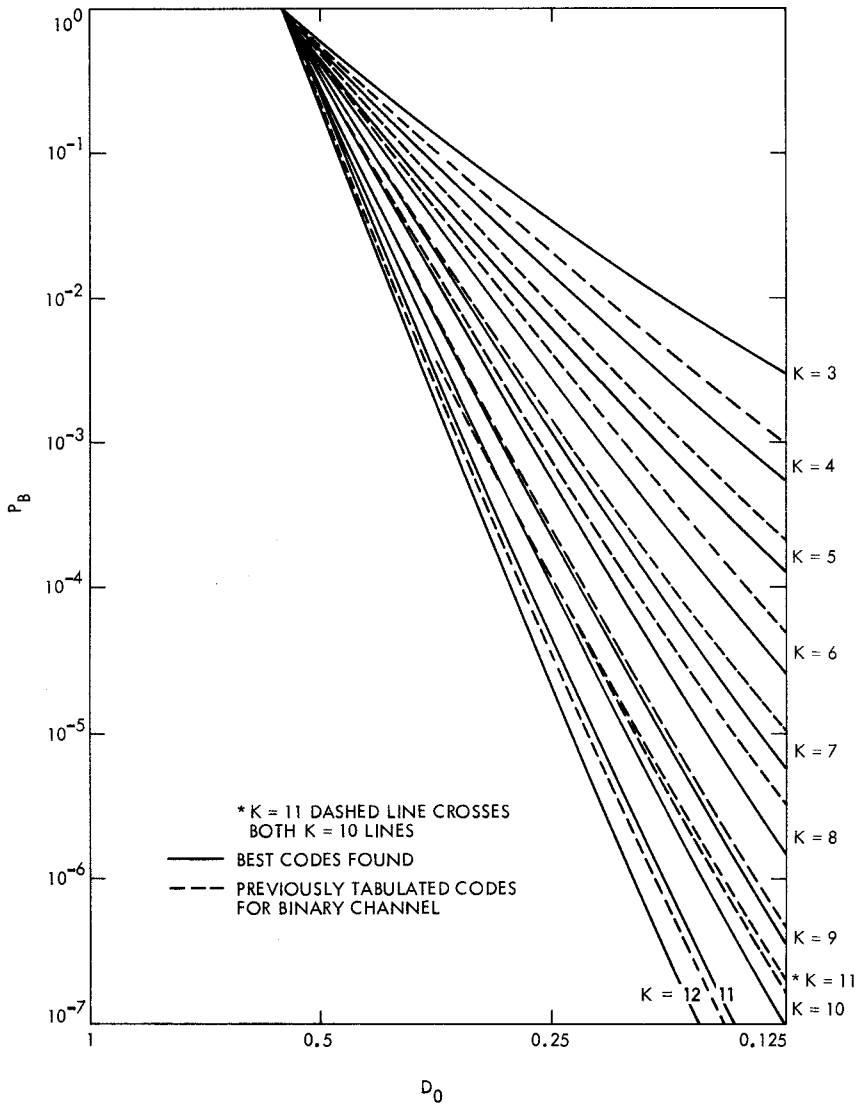

Fig. 2. Estimates of P_B vs D_0 for rate 1/2 codes on 4-ary channel

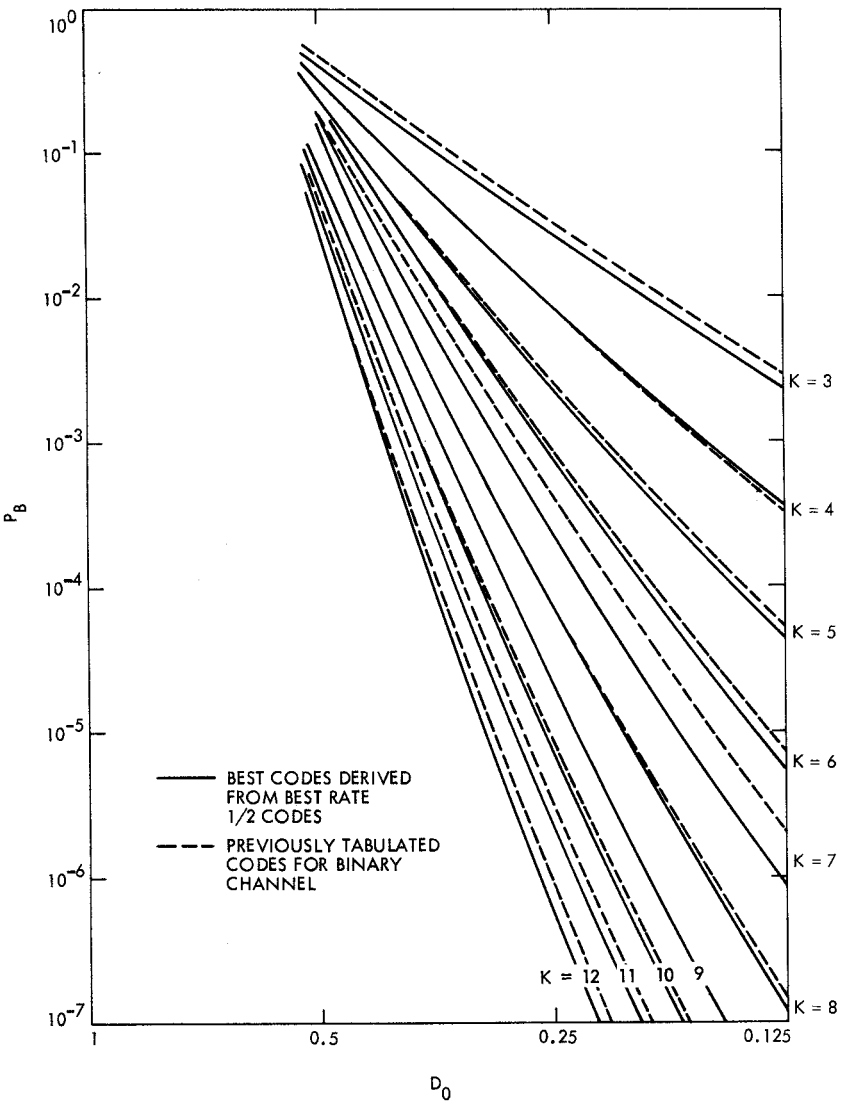

Fig. 3. Estimates of P_B vs D_0 for rate 1/3 codes on 8-ary channel

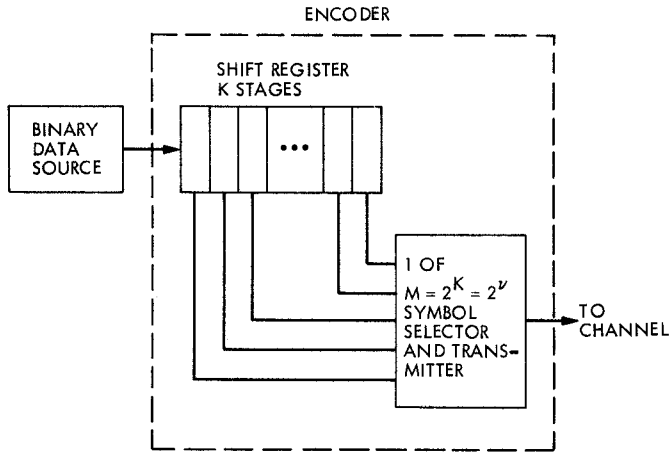

Fig. 4. The best convolutional codes of constaint length *K* on the *M*-ary orthogonal channel, achieved by allowing ν to increase to $\nu = K$, while $M = 2^\nu$

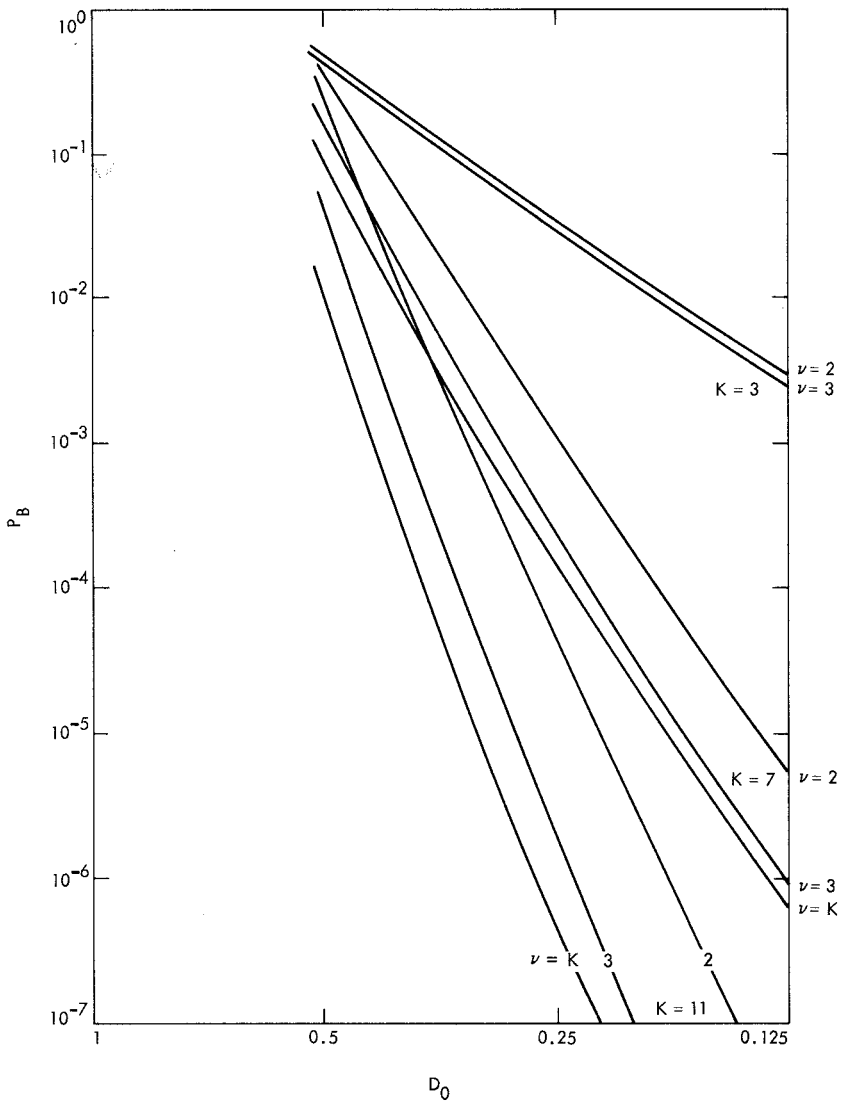

Fig. 5. Dependence of best P_B vs D_0 curves on ν, with $M = 2^\nu$, for $K = 3, 7, 11$

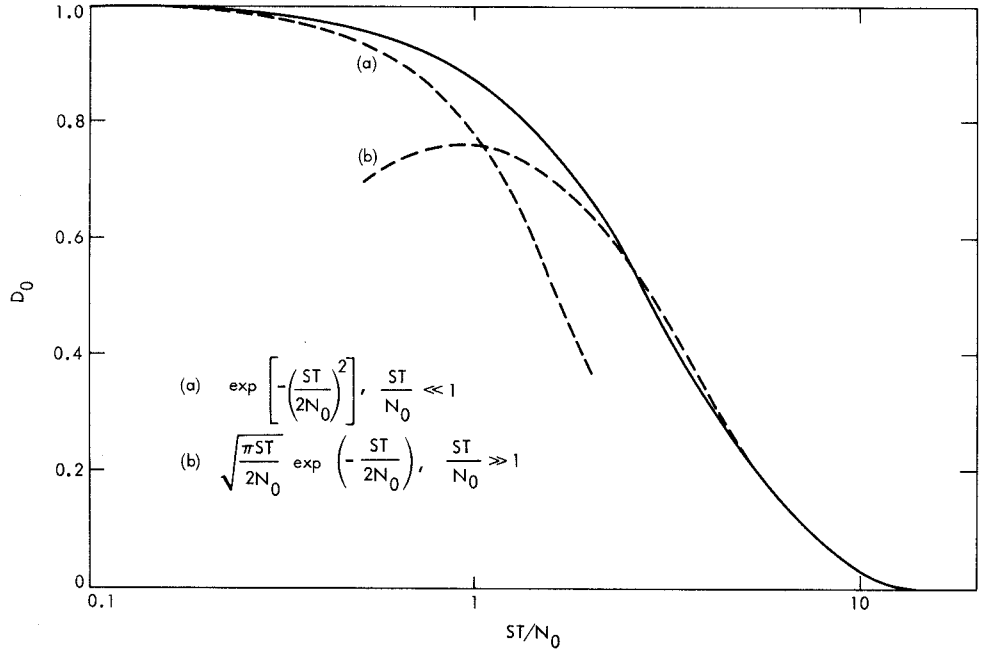

Fig. 6. D_0 vs signal-to-noise ratio ST/N_0 for noncoherent MFSK and the two asymptotic approximations

Appendix

```
    1: C   SUBROUTINE TO EVALUATE CONVOLUTIONAL CODES
    2: C   R F LYON    JPL 238-420   SEC 331   X2766
    3: C
    4: C   LG(1)  FIRST CONNECTION VECTOR
    5: C    - - -
    6: C   LG(N2) LAST CONNECTION VECTOR
    7: C   K      CONSTRAINT LENGTH
    8: C   N1/N2  RATE IN BITS/SYMBOL
    9: C   M      MINIMUM DISTANCE
   10: C   LM     MAXIMUM INPUT LENGTH TO TEST
   11: C   FLAG   PRINT CONTROL
   12: C          FLAG>0 PRINT ALL COEFFS OF T(D,N,L)
   13: C          FLAG<0 PRINT ALL COEFFS OF T(D,N)
   14: C          FLAG=0 PRINT HEADER ONLY
   15: C   DIST   EXTERNAL SUBROUTINE DEFINING DISTANCE MEASURE
   16: C   DISB   EXTERNAL SUBROUTINE GIVING A DISTANCE BOUND
   17: C   ERR    EXTERNAL SUBROUTINE TO ESTIMATE ERROR PROBABILITY
   18: C   EROUT  ENTRY POINT TO PRINT ERROR ESTIMATES
   19:        SUBROUTINE EVAL(LG,K,N1,N2,M,LM,FLAG)
   20:        DIMENSION KOUNT(12,12,15),KSUM(12,15)
   21:        DIMENSION LG(4),LC(4),ND(4)
   22: C      CLEAR COUNTERS TO BE USED
   23:        KMIN=0
   24:        DO 1 L=1,LM
   25:        DO 1 N=1,L
   26:        CALL DISB(NDB,K,L,N1)
   27:        DO 1 ND1=M,NDB
   28: C      COEFFICIENT OF T(D,N,L)
   29:      1 KOUNT(L,N,ND1-M+1)=0
   30:        DO 21 N=1,LM
   31:        DO 21 ND1=M,NDB
   32: C      COEFFICIENT OF T(D,N)
   33:     21 KSUM(N,ND1-M+1)=0
   34: C      MINIMUM DISTANCE INITIALIZE
   35:        MD=1000000
   36: C      LOOP ON INPUT LENGTHS UP TO LM
   37:        DO 11 L=1,LM
   38: C      LOOP ON INPUTS OF LENGTH L
   39:        DO 10 IN=IOR(2**(L-1),1),2**L,2
   40: C      WEIGHT INITIALIZE
   41:        N=0
   42: C     CONVOLVE INPUT WITH CODE GENERATORS
   43: C      LOOP ON N2 CODE GENERATORS
   44:        DO 2 I=1,N2
   45: C      CLEAR SPACE FOR CONVOLUTION
   46:      2 LC(I)=0
   47: C      LEFT JUSTIFY INPUT
   48:        II=ISL(IN,31-L)
   49: C      LOOP ON CODE GENERATORS
   50:      3 DO 4 I=1,N2
   51: C      SHIFT OUTPUTS
   52:      4 LC(I)=ISL(LC(I),1)
   53: C      SHIFT INPUT
   54:        II=ISL(II,1)
```

```
C     TEST OF LEFTMOST BIT
      IF(II) 5,7,3
C     LOOP ON GENERATORS
    5 DO 6 I=1,N2
C     ADD GENERATORS TO OUTPUTS, MOD 2
    6 LC(I)=IEOR(LC(I),LG(I))
C     COUNT INPUT WEIGHT
      N=N+1
      GO TO 3
C     EVALUATE DISTANCES OF CONVOLVED SEQUENCES FROM ZERO
    7 CALL  DIST(ND,LC,N1,N2)
C     LOOP ON SHIFTED VERSIONS OF INPUT
      DO 8 I=1,N1
C     UPDATE NIMIMUM DISTANCE
      IF(ND(I).LT.MD) MD=ND(I)
C     ABORT IF DISTANCE TOO SHORT
    8 IF(ND(I).LT.M) RETURN
C     LOOP ON SHIFTED INPUTS
      DO 9 I=1,N1
      ND1=ND(I)-M+1
C     INCREMENT COUNTERS
      KSUM(N,ND1)=KSUM(N,ND1)+1
      KOUNT(L,N,ND1)=KOUNT(L,N,ND1)+1
      IF(ND1.EQ.1) KMIN=KMIN+1
    9 CONTINUE
C     MAKE INCREMENTAL CHANGE IN ERROR ESTIMATES
      CALL ERR(N,ND,N1)
   10 CONTINUE
C     FINISH LOOPS ON INPUTS
   11 CONTINUE
C     PRINT HEADER
      PRINT 100,FLAG,K,N1,N2,N2,(I,LG(I),I=1,N2),MD,KMIN
C     CHECK PRINT CONTROL
      IF(FLAG) 13,16,15
C     LIST ALL COUNTERS
   15 DO 12 L=1,LM
      CALL DISB(NDB,K,L,N1)
      PRINT 101,L,NDB-MD+1,(ND1,ND1=MD,NDB)
      DO 12 N=MIN(L,2),L
      PRINT 102,N,(KOUNT(L,N,ND1-M+1),ND1=MD,NDB)
   12 CONTINUE
C     LIST COUNTERS WITHOUT LENGTH FACTOR
   13 CONTINUE
X     PRINT 101,LM,NDB-MD+1,(ND1,ND1=MD,NDB)
X     DO 14 N=1,LM
X  14 PRINT 102,N,(KSUM(N,ND1-M+1),ND1=MD,NDB)
C     PRINT ERROR ESTIMATES
      CALL EROUT
C     RETURN MINIMUM DISTANCE
   16 M=MD
      RETURN
  100 FORMAT(I1,'K=',I2,'  RATE',I2,'/',I1,N(' G',I1,'=',Z3),'  DF=',I2,I5)
  101 FORMAT('OL**',I2,X, N(2X,'D**',I2))
  102 FORMAT(' N**',I2,15(I7))
      END
```

```
      C     SUBROUTINE TO COMPUTE UNION BOUND ESTIMATE OF PE AND PB
      C     N    INPUT WEIGHT (EXPONENT OF N IN T(D,N))
      C     N1   NUMBER OF SHIFTED VERSIONS OF INPUT
      C     ND   SET OF DISTANCES (EXPONENTS OF D IN T(D,N))
      C          FOR EACH SHIFTED VERSION OF INPUT
            SUBROUTINE ERR(N,ND,N1)
            DIMENSION ND(4),DO(8),PE(8),PB(8)
      C     PE= T(DO)
      C     PB= DT(DO,N)/DN|N=1
            DO 1 I=1,N1
            DO 1 J=1,K
            PB(J)=PB(J)+N*DO(J)**ND(I)
            PE(J)=PE(J)+  DO(J)**ND(I)
          1 CONTINUE
            RETURN
      C     ENTRY TO SET UP VALUES OF DO AND CLEAR PB AND PE
      C     KK   NUMBER OF VALUES TO DO SUPPLIED
      C     DD   VALUES OF DO,STARTING AT INDEX 1
            ENTRY ERSET(DD,KK)
            DIMENSION DD(8)
            K=KK
            DO 2 J=1,K
            DO(J)=DD(J)
            PB(J)=0
            PE(J)=0
          2 CONTINUE
            RETURN
      C     ENTRY TO PRINT RESULTS
            ENTRY ERROUT
            PRINT 4
      C     THREE COLUMNS: DO, PE, PB
      C     FOR K VALUES OF DO
            DO 3 J=1,K
            PRINT 5,DO(J),PE(J),PB(J)
          3 CONTINUE
            RETURN
          4 FORMAT('0     DO        PE          PB')
          5 FORMAT(F8.4,2F9.7)
            END
```

```
1: C      SUBROUTINE DEFINING DISTANCE MEASURE ON M-ARY SYMMETRIC CHANNEL
2: C      ND  LIST OF N1 DISTANCES TO RETURN
3: C      LC  LIST OF N2 CODER OUTPUTS
4: C      N1/N2  CODE RATE
5: C      ONLY ONE OF EACH N1 OUTPUT POSITIONS IS ACTUALLY OUTPUT FOR EACH
6: C       OF N1 SHIFTED VERSIONS OF THE INPUT
7:        SUBROUTINE DIST(ND,LC,N1,N2)
8:        DIMENSION ND(4),LC(4)
9:        L=0
10: C     DIST= NUMBER OF PLACES IN WHICH ANY OF N2 OUTPUTS ARE ONE
11:       DO 1 I=1,N2
12: C     'OR' OUTPUTS TOGETHER
13:     1 L=IOR(L,LC(I))
14:       DO 2 I=1,N1
15:     2 ND(I)=0
16:     3 DO 5 I=1,N1
17:       L=ISL(L,1)
18: C     CHECK LEFTMOST BIT
19:       IF(L) 4,6,5
20:     4 ND(I)=ND(I)+1
21:     5 CONTINUE
22:       GO TO 3
23:     6 RETURN
24: C     ENTRY TO COMPUTE DISTANCE BOUND ON M-ARY SYMMETRIC CHANNEL
25:       ENTRY DISB(NDB,K,LL,N1)
26: C     BOUND IS INTEGER>=(K+L-1)/N1
27: C     I.E. LENGTH OF OUTPUT SEQUENCE
28:       NDB=(K+LL+N1-2)/N1
29:       RETURN
30:       END
```